Catching Transparent Phish: Analyzing and Detecting MITM Phishing Toolkits

Brian Kondracki
Stony Brook University
bkondracki@cs.stonybrook.edu

Babak Amin Azad
Stony Brook University
baminazad@cs.stonybrook.edu

Oleksii Starov
Palo Alto Networks
ostarov@paloaltonetworks.com

Nick Nikiforakis
Stony Brook University
nick@cs.stonybrook.edu

ABSTRACT

For over a decade, phishing toolkits have been helping attackers automate and streamline their phishing campaigns. Man-in-the-Middle (MITM) phishing toolkits are the latest evolution in this space, where toolkits act as malicious reverse proxy servers of online services, mirroring live content to users while extracting credentials and session cookies in transit. These tools further reduce the work required by attackers, automate the harvesting of 2FA-authenticated sessions, and substantially increase the believability of phishing web pages.

In this paper, we present the first analysis of MITM phishing toolkits used in the wild. By analyzing and experimenting with these toolkits, we identify intrinsic network-level properties that can be used to identify them. Based on these properties, we develop a machine learning classifier that identifies the presence of such toolkits in online communications with 99.9% accuracy.

We conduct a large-scale longitudinal study of MITM phishing toolkits by creating a data-collection framework that monitors and crawls suspicious URLs from public sources. Using this infrastructure, we capture data on 1,220 MITM phishing websites over the course of a year. We discover that MITM phishing toolkits occupy a blind spot in phishing blocklists, with only 43.7% of domains and 18.9% of IP addresses associated with MITM phishing toolkits present on blocklists, leaving unsuspecting users vulnerable to these attacks. Our results show that our detection scheme is resilient to the cloaking mechanisms incorporated by these tools, and is able to detect previously hidden phishing content. Finally, we propose methods that online services can utilize to fingerprint requests originating from these toolkits and stop phishing attempts as they occur.

CCS CONCEPTS

• **Security and privacy → Phishing**.

KEYWORDS

phishing; web security; social engineering

ACM Reference Format:
Brian Kondracki, Babak Amin Azad, Oleksii Starov, and Nick Nikiforakis. 2021. Catching Transparent Phish: Analyzing and Detecting MITM Phishing Toolkits. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS '21), November 15–19, 2021, Virtual Event, Republic of Korea.* ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3460120.3484765

1 INTRODUCTION

The combination of a username and password is the default gatekeeper to nearly all online services that users interact with, on a daily basis. It is therefore no surprise that this sensitive information is in high demand among malicious actors who go to great lengths to obtain it, in order to access sensitive information and act on behalf of victims. One of the most commonly used methods of acquiring this information is through social engineering, in the form of *phishing*. Phishers impersonate trustworthy entities in an attempt to lure victims into disclosing private information, such as account credentials and banking information.

Traditionally, phishing websites were hosted entirely on attacker-owned and compromised web servers where attackers would host realistic-looking copies of their target websites in hope of convincing users to disclose their credentials. These credentials were stored either on the original server or communicated to the attacker (e.g. via an email) for later abuse [57]. These rudimentary phishing setups required substantial effort on behalf of attackers to clone target websites, make the necessary content-modification to make these sites operational, and repeat this entire process to match updates to the UI of the target website.

To reduce the effort required by attackers to create and serve phishing content, all-in-one *phishing toolkits* began to overtake traditional setups. These toolkits revolutionized how phishing websites are created by automatically fetching static copies of web pages from targeted websites, serving them to victims, and preventing detection through cloaking mechanisms—all while requiring minimal effort by attackers [50]. However, the increasing adoption of two-factor-authentication (2FA) mechanisms by online services, and the rapid evolution of web content has increased the need for phishing toolkits to adopt real-time mechanisms in place of antiquated static content.

These limitations fueled the proliferation of a new generation of *Man-in-the-Middle (MITM)* phishing toolkits [8, 14, 15]. These next-generation phishing toolkits act as malicious reverse proxy servers, forwarding requests and responses between the victim and the target web server, while extracting credentials and session cookies in transit. This eliminates the need to create and maintain realistic phishing web pages (the phishing page is now a "perfect" copy of the victim website) as well as manually communicating with the target website to send the user credentials and 2FA codes to obtain the authenticated session cookie. Moreover, because of the continuous proxying of requests and responses, these tools greatly increase the believability of the attack by allowing users to continue browsing the phishing site *after* they authenticate, as if they are truly interacting with the target site.

In this work, we present the first analysis of this new generation of phishing toolkits. We study 13 versions of popular MITM phishing toolkits and present a methodology to fingerprint them in the wild, both from the perspective of a user interacting with a phishing website, as well as the target website receiving login attempts from a toolkit impersonating a regular user. We produce a globally-diverse dataset of laboratory MITM phishing toolkit deployments, detailing their network-level characteristics. Using this dataset, we develop a machine learning classifier that leverages the network timing discrepancies inherent to reverse proxy servers to detect the presence of MITM phishing toolkits with 99.9% accuracy. We show that our classifier is robust to changes made by attackers to thwart fingerprinting attempts, and we demonstrate the ability of our classifier to detect unseen phishing toolkits. By proposing fingerprinting methods that uniquely identify MITM phishing toolkits, we enhance the ability of web-service providers to pinpoint malicious login requests and flag them before authentication is completed.

To automate the discovery and analysis of MITM phishing toolkits on the web, we create a fingerprinting tool which we call PHOCA. PHOCA can be directly integrated into current web infrastructure such as phishing blocklist services to expand their coverage on MITM phishing toolkits, as well as popular websites to detect malicious requests originating from MITM phishing toolkits.

Using PHOCA, we study the usage trends of these tools in the wild over the course of a year, discovering 1,220 websites utilizing MITM phishing toolkits targeting popular services including Google, Yahoo, Twitter, and Facebook. We observe that, due to their highly-targeted nature and cloaking mechanisms, MITM phishing toolkits occupy a blind spot in current phishing blocklists, as only 43.7% of domains and 18.9% of IP addresses associated with the MITM phishing toolkits we discovered appear on popular blocklists. Additionally, through our collaboration with Palo Alto Networks, we find that enterprise users are targeted by MITM phishing toolkits, with 260 of our discovered phishing sites receiving 6,403 customer requests over a six-month period.

In summary, the contributions of this paper are as follows:

- We present the first, in-depth study of MITM phishing toolkits.
- We propose a machine learning classifier that utilizes network-level features to classify phishing websites hosted by such toolkits with 99.9% accuracy.
- We develop a MITM phishing toolkit fingerprinting framework, called PHOCA, that can collect data on and classify MITM phishing toolkits on the web.
- We use PHOCA to explore the use of MITM phishing toolkits in the wild and find that current phishing blocklists do not effectively report these malicious websites.
- We show how these toolkits can be identified from both the perspective of the victim user and target web server.

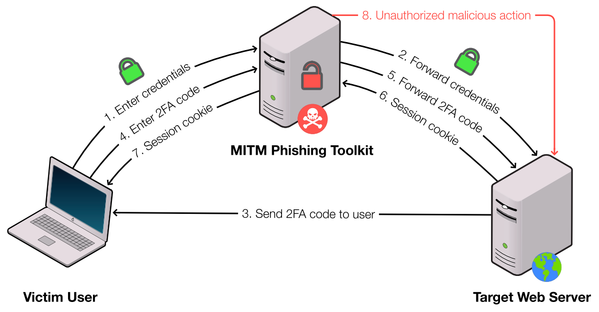


Figure 1: Architecture of MITM phishing toolkits.

2 BACKGROUND

In this section, we provide background information on MITM phishing toolkits and the threat model we consider in this paper.

2.1 Reverse Proxy Servers

Reverse proxy servers are front-end web servers that perform all direct communication with each web client. They are utilized in a variety of use cases from load balancing, to providing authentication for services residing on private networks. These servers act as "middlemen," brokering connections between users and backend web servers. Typically, TLS connections are terminated at reverse proxy servers, thereby decreasing the complexity of configuring TLS certificates for website administrators by creating a single configuration point. Some of the most popular reverse proxy servers today are: Squid [22], Nginx [17], and Apache Traffic Server [2].

2.2 MITM Phishing Toolkits

MITM phishing toolkits function as reverse proxy servers between victims and one or more target web servers. These toolkits act as web servers when communicating with victims, and clients when communicating with target web servers. Figure 1 shows the general architecture of MITM phishing toolkits.

This design lends itself to increased believability of the phishing attack since the returned web pages are live on the target web server and thus *indistinguishable* to the victim. Additionally, unlike traditional phishing attacks, where believable behavior ceases after the desired information (e.g. credentials and credit card numbers) is acquired, these toolkits persist the victim's browsing session after authentication is complete. This means users can browse the target website with their authenticated session *through* the phishing server. This puts the victim at ease and increases the timespan that the session cookie is valid, allowing the attacker more time to conduct their desired malicious actions.

One powerful use of these toolkits is to compromise user accounts that are protected by 2FA mechanisms. When credentials are provided by victims, they are simply read and saved for later use before being forwarded to the target web server. The target web server will then either send a 2FA code to the user through a separate, pre-established communication channel (such as SMS text and email) or rely on a mobile app/hardware token to generate such a code at the client side. In both cases, the client then submits that code to the phishing toolkit, where it is again forwarded to the

Table 1: Comparison of network-level to application-level fingerprinting in response to popular cloaking techniques. ✓/✗ indicates the fingerprinting technique would succeed or fail against the specified cloaking technique, respectively.

Cloaking Type	Application-Layer	Network-Layer
URL Obfuscation	✗	✓
User Interaction	✗	✓
Fingerprinting	✗	✓
Bot Activity	✗	✓
IP Address	✗	✗

target web server. When authentication is completed, the session cookie provided by the target web server is saved by the MITM phishing toolkit, enabling attackers to now send authenticated requests in the name of the victim.

2.3 Threat Model

Like a typical phishing attack, attackers who utilize a MITM phishing toolkit need to deploy it on a web server and send a link to that web server to their potential victims.

As Figure 1 shows, victims communicate with the phishing server over an HTTPS connection. Even though in theory, an HTTPS connection between the user and the phishing server is not necessary, modern web browsers show a barrage of warnings for websites that are visited over HTTP, particularly when the user is providing input in HTML forms. Therefore, all of the MITM phishing toolkits that we study in this work use valid TLS certificates for their phishing pages. *All* requests made to the phishing server are forwarded along to the target web server, including malformed requests as well as requests towards non-existing resources. Connections between the phishing server and target web server are made over an additional HTTPS connection, where the phishing server takes the role of a web client. All traffic between the victim and target web server is available in cleartext to the attacker.

Due to the server-side cloaking mechanisms utilized by MITM phishing toolkits, only intended victims see malicious content. This thwarts all content-based phishing detectors, which require access to the phishing content. We note that organizations with inline access to network communications can view phishing content targeted at their users. However, due to attackers' complete control over the data in the application layer, content-based phishing detection from this vantage point is still prone to failure. We therefore must ensure that all methods employed to detect the presence of these toolkits do not depend upon the integrity of proxied data.

Table 1 demonstrates the effectiveness of network-layer and application-layer fingerprinting against cloaking techniques used by attackers today [38, 50, 51]. As application-layer fingerprinting requires access to phishing content, it is thwarted by all widely-used cloaking categories. Network-layer fingerprinting, on the other hand, analyzes features of the network connection and web server in question, making it effective against all categories except IP-based cloaking. However, it should be noted that IP-based cloaking would bypass any form of detection/fingerprinting that originates from an IP address considered to be suspicious from the point of view of an attacker.

3 MITM PHISHING TOOLKIT CLASSIFIER

In this section, we first describe the three MITM phishing toolkits that we evaluate and the functionality that they offer to attackers. We then provide the details of the training and validation process used to create a classifier for detecting these toolkits.

3.1 MITM Phishing Toolkit Identification and Collection

Prior to studying MITM phishing toolkits, it is important to decide upon a definition that accurately describes their functionality. For the scope of this paper, we define a MITM phishing toolkit as a reverse proxy server that mirrors a target web page to a victim while harvesting credentials, 2FA codes, and web page content in transit. The important distinction between these toolkits and other phishing tools is the continuous proxying of user traffic to and from the target web server, pre- and post-authentication.

Using this definition, we searched for all MITM phishing toolkits on popular hacker forums and code repositories, both on the clear web as well as on Tor [24]. Through this search, we identified three MITM phishing toolkits: *Evilginx* [8], *Muraena* [15], and *Modlishka* [14]. We also discovered similar toolkits such as *CredSniper* [6] and *Reelphish* [20]. However, as these do not proxy user traffic to the target web server (i.e. they do not act as MITM phishing toolkits), we consider them as out of scope for this work.

3.2 MITM Phishing Toolkit Functionality

While all current MITM phishing toolkits have a similar architecture and share common goals, there are differences in their feature-sets that influence their popularity among attackers.

Evilginx. Of the toolkits studied in this paper, Evilginx is the easiest to operate due to its command-line interface which is used to configure most aspects of the phishing server. Additionally, along with hosting a web server, Evilginx also hosts its own DNS server and automatically creates all TLS certificates needed using the Let's Encrypt [12] API. This significantly lowers the barrier of entry for attackers, allowing even the least technically adept to launch their own phishing campaigns. Evilginx is also the only tool of the three studied that allows attackers to host multiple phishing pages simultaneously. Each individual phishing page responds to a subdomain of the primary domain provided by the attacker. Code listing 1 in the Appendix shows an example configuration file for Evilginx.

Evilginx allows attackers to launch highly targeted attacks, and cloak their actions from anyone but the intended victim. This cloaking is accomplished by generating tokenized URLs, referred to internally as *lures*. These unique URL parameters must be included in requests to view the phishing content. An example of a tokenized URL generated by Evilginx is as follows: *https://evil.com/xICcxSqs*, where the phishing domain name is followed by a random token. All requests missing valid tokens are redirected to a web page of the attacker's choice. Attackers can also quickly disable individual tokens, preventing even tokenized URLs from responding with phishing content. These features effectively cloak the presence of Evilginx, increasing the uptime of campaigns before they are detected and subsequently halted through blocklisting.

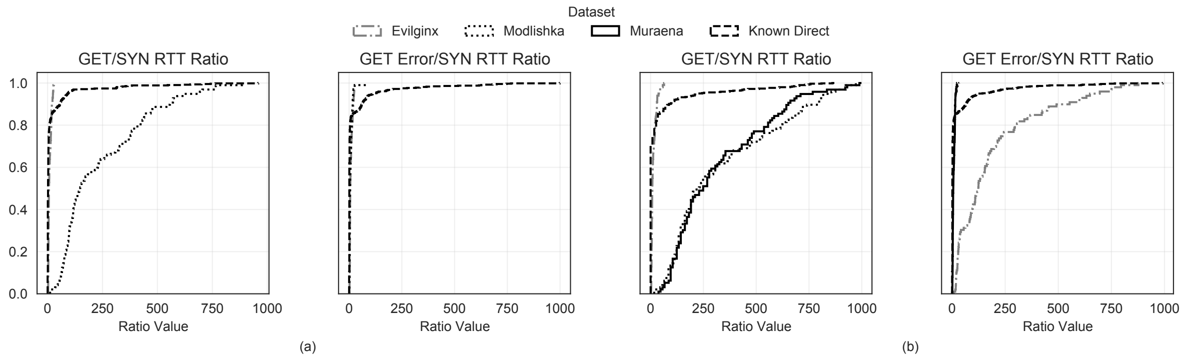


Figure 2: Ratio of TCP SYN/ACK to valid and malformed HTTP GET Request RTTs over (a) HTTP and (b) HTTPS. (Muraena is not present in plot (a) since it does not respond to HTTP requests.)

This diverse feature-set has led to a surge in the popularity of this tool among attackers, and an increased attention from anti-virus services. According to VirusTotal, eight different antivirus products mark Evilginx as malicious [26]. This finding demonstrates the dual nature of these types of toolkits where the very same tools that can be used in the context of a legal penetration-testing engagement, are also used by attackers when they break into servers.

Modlishka. Modlishka is a bare bones approach to a MITM phishing toolkit. Unlike Evilginx, it is limited to targeting one domain at a time and does not provide a command-line interface for configuration. Also, it is up to the attacker to provide a certificate, as Modlishka is only capable of automatically generating self-signed certificates. Modlishka can also be configured to remove all encryption and security headers from requests to target web servers.

Muraena. While Evilginx and Modlishka require the attacker to manually create configuration files detailing the domains and HTML attributes that should be replaced or removed from requests and responses, Muraena automates this process using a web crawler. Additionally, Muraena automates post-compromise actions that an attacker would want to execute such as, changing passwords and exfiltrating data. This is done using a companion tool called *Necrobrowser* [16]. This tool takes the session cookies extracted using Muraena and launches an instrumented Chrome instance using the Chrome DevTools Protocol, to perform the desired malicious actions on the target website without attacker intervention, greatly increasing the effectiveness of phishing campaigns.

3.3 Exploratory Data Analysis

As described in Section 2.3, the unique architecture of MITM phishing toolkits allows attackers to create impersonating web pages that effectively fool victims into providing their credentials. However, this architecture also introduces discrepancies in packet round-trip-times (RTTs), enabling the fingerprinting of these toolkits at the network level. As two distinct HTTPS sessions must be maintained to broker communication between the victim user and target web server, the ratio of various packet RTTs, such as a TCP SYN/ACK request and HTTP GET request, will be much higher when communicating with a reverse proxy server than with an origin web server directly [45]. This ratio is further magnified when the reverse proxy server intercepts TLS requests, which holds true for MITM phishing toolkits. Intuitively, this can be attributed to HTTP requests propagating through to the target web server, while TCP SYN packets are answered directly at the MITM phishing toolkit.

To verify the efficacy of these timing discrepancies in identifying MITM phishing toolkits, we perform a series of exploratory measurements. We record the RTTs of TCP SYN/ACK packets and HTTP GET requests to each toolkit as well as an Apache [3] web server under our control. We make both valid and malformed requests in order to entice a direct response from the toolkits rather than a proxied response from the target web server. The results from this experiment are shown in Figure 2. The four cumulative distribution functions represent the ratio of TCP SYN/ACK requests to valid and malformed GET requests made over HTTP and HTTPS. Here, smaller ratio values indicate that the two requests are answered by the same physical machine, while larger ratio values imply responses to GET requests are sent by a machine at least one hop away from the machine that responded to the TCP SYN request (i.e. an origin server situated behind a reverse proxy). We find that each of the MITM phishing toolkits can be clearly distinguished from the direct-server distribution in at least one of the four evaluated RTT ratios.

3.4 MITM Phishing Toolkit Classifier

Motivated by the results from our exploratory analysis, we develop a machine-learning-based classifier trained on data gathered from real-world websites and each of the MITM phishing toolkits in a laboratory setting.

Feature Engineering

As mentioned in Section 2.3, all content viewed on the client device is at the complete control of the attacker, making any features present in the application layer easily modifiable. Thus, when designing our classifier, we focus on features inherent to the nature of the man-in-the-middle architecture present in the toolkits. Using these types of features provides us with a robust and powerful classifier that is not only effective at the time of writing, but is also adaptable to changes in existing tools as well as future tools.

To this end, we divide our feature set into *Network Timing Features*, and *TLS Library Features*.

• **Network Timing Features:** As described in Section 3.3, we use the RTT ratios of various points in TCP and TLS handshakes, as well as HTTP GET requests. We make both valid and malformed HTTP requests in order to solicit proxied and direct responses respectively from MITM phishing toolkits.

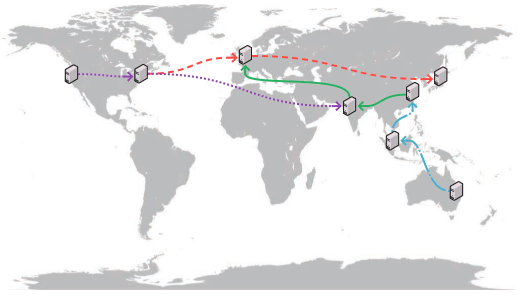


Figure 3: Architecture of experimental framework used to collect network timing data of MITM phishing toolkits.

• **TLS Library Features:** Since MITM phishing toolkits typically do not use the same web or reverse proxy server software as benign websites, they make use of different TLS libraries to handle HTTPS connections from clients. We therefore use TLS implementations as a distinguishing factor. We treat each TLS version supported by the current web server as a binary feature in our classifier. Additionally, we use the TLS fingerprinting tool TLS Prober [23] to identify the TLS library utilized by the current web server based on the format of TLS packets it transmits. TLS Prober determines the library used by a web server via a series of TLS `Client Hello` packets. By analyzing the format of each server response, the TLS library implementation can be determined. TLS Prober returns a map of TLS libraries to the probability each library is used by the web server in question. We treat each potential TLS library, and its associated probability, as a numeric feature in our classifier. Since TLS Prober does not make predictions off of configurable options, such as cipher suites, an attacker attempting to bypass this fingerprinting would need to re-engineer the entire TLS integration of a MITM phishing toolkit and somehow mimic the TLS stack of a real web server.

By heavily relying on features deeply embedded into the architecture of MITM phishing toolkits, our classifier is largely future-proof and toolkit-agnostic. More specifically, of the 199 features our classifier is composed of, 14 are network timing features, and 185 are TLS library features. Table 7 located in the Appendix shows our full list of features.

Dataset Collection

Websites on the Internet are served from a variety of different network architectures. More specifically, client requests are routed either directly to web servers, or through reverse proxies in the form of load balancers and CDNs. To effectively distinguish MITM phishing toolkits from benign websites of these categories, we collect ground-truth data from each of the following groups:

1. Non-Proxied Web Pages: To gather a list of websites with the highest likelihood of being served directly by a web server without an intervening proxy server, we use a heuristic of domains pointing to IP addresses of the cloud-hosting providers Digital Ocean [7] and Linode [13], as well as websites of local small businesses. The aforementioned cloud-hosting providers are popular because they offer simple virtual private servers, as opposed to the vast array of products and services (including CDNs and load balancers) that larger cloud-hosting providers (such as AWS and Google Cloud) offer. We therefore reason that websites hosted on these platforms are most likely hosted directly by origin servers. Similarly, because

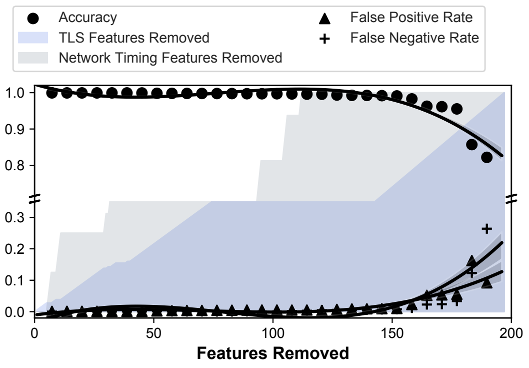


Figure 4: Accuracy, False Positive, and False Negative rates of classifier when iteratively removing most important features.

of the low traffic volume that the websites of local businesses attract (such as local restaurants) we argue that they are highly unlikely to be paying for load balancing and CDNs.

2. Reverse Proxy Web pages: We utilize the public IP address range of Cloudflare [5] (one of the most popular anti-DDoS, CDN services) to curate a list of benign websites hosted behind a reverse proxy server. We use reverse DNS lookups on each IP address in the Cloudflare subnet to gather a list of Cloudflare-managed domains.

3. MITM Phishing Toolkit Web pages: As the RTT of requests will vary across different geographic distances, it is important to collect network timing data from a large number of vantage points. To accomplish this, we design a data collection methodology modeled after the work of Alexander [29]. More specifically, we launch 30 globally distributed nodes hosted on AWS [1], where each node hosts a web client, the three phishing toolkits, and an Apache web server simultaneously. This globally distributed infrastructure, visualized in Figure 3, allows us to record network timings for many of the potential geographic distributions of victim⟶phishing toolkit⟶target web server permutations. For example, our infrastructure allowed us to obtain measurements that included the three parties (victim, phishing toolkit, and target web server) all being located in North America, as well as the victims being located in Asia, having their traffic proxied by an EU-residing phishing server, onto a US-based target web server.

In each permutation, `Node A` sends an HTTP GET request to the port of a phishing toolkit on `Node B`, where it is then forwarded to the Apache web server hosted on `Node C`. Since it is unlikely for a phishing toolkit to exist on the same host as a target web server or victim user, we exclude these scenarios. This leaves us with $n(n-1)(n-2)$, or 24,360 permutations per toolkit. Since we record data on three toolkits, we obtain a total of 73,080 network measurements for the evaluated MITM phishing toolkits.

We compile a ground-truth dataset composed of all facets of each tool's requests and responses as well as benign websites. In total, we collected 73,080 network requests from the MITM toolkit and benign categories, for a total of 146,160 data points.

3.5 Model Training and Validation

Using our compiled ground truth data, we constructed training and testing datasets with a 1:1 phishing-to-benign ratio. We then trained a Random Forest classifier with a minimum sample split of 2 and 100

Table 2: Model performance when training on oldest release of each MITM phishing toolkit and testing on all subsequent releases.

	Accuracy	Precision	Recall
A. Restrict classifier to individual feature groups			
Network Timing	98.5%	98.5%	98.5%
TLS	99.9%	99.9%	99.9%
Total	99.9%	99.9%	99.9%
B. Training on oldest release, testing on newer releases			
Evilginx v2.0	98.6%	100%	97.9%
Muraena v0.1	87.7%	100%	85%
Modlishka v1.0.0	99.8%	100%	99.8%
C. Exclude specified toolkit data from training dataset			
Evilginx	97.5%	100%	97.5%
Muraena	96.4%	100%	96.4%
Modlishka	99.8%	100%	99.8%

estimators. We empirically determined that these hyperparameters provide us with the highest accuracy as well as lowest false positive and negative rates. We chose a Random Forest classifier because of its proven track record for security-related applications [30, 33, 48, 54, 56] while still maintaining a level of explainability, not found in other types of machine-learning classifiers. We achieve an accuracy score of 99.9% and a five-fold cross validation score of 99.9%.

Model Feature Importance

To ensure our classifier does not overfit on a small subset of powerful toolkit-specific features, we study the decrease in accuracy as well as the increase in false positive and false negative rates as we iteratively remove the most important features.

Figure 4 demonstrates the decay in model effectiveness while iteratively removing the most important feature and retraining. The shaded regions represent the percentage of features from each of the TLS and network timing categories removed throughout the experiment. We observe that this feature removal does not have significant effects on our classifier as the accuracy remains above 97% even after removing the top 150 most important features. Further, Table 2A shows the performance when restricting the classifier to only one of the feature groups at a time (e.g. training using just network-timing features and ignoring TLS). Our classifier continues to perform well with each feature group isolated, demonstrating that it is not dependent upon a small group of features, but rather the entire ensemble of features as a whole. As a result, we argue that even if attackers are aware of our tool's presence, it will not be trivial to evade detection by selectively patching individual features.

Model Generalizability

Utilizing network-level features increases the robustness of our classifier against many of the modifications attackers can make to thwart fingerprinting attempts. However, we seek to create a classifier that is not only effective in detecting the three identified toolkits in their current form, but is also generalizable to updates to the existing toolkits as well as toolkits created in the future.

We test the effect of MITM phishing toolkit updates on model performance by downloading all releases of the three MITM phishing toolkits, dating up to two years old, and collecting data on each using the same methodology described in Section 3.4. We then train our classifier on the oldest release available of each toolkit along with known non-MITM phishing toolkit data, and test on the remaining releases. In total, we collected 13 MITM phishing toolkit versions with Evilginx, Muraena, and Modlishka having four, seven, and two versions respectively. The results of this experiment are shown in Table 2B. We find that, generally, the network-level features of each toolkit remain constant through incremental updates. The only exception to this is Muraena, which included support for HTTP requests in the most recent release (version 0.3) with prior releases only answering HTTPS requests. This discrepancy in behavior leads to the observed dropoff in performance. However, this experiment represents a worst-case-scenario for defenders where the classifier is not updated to match changes in the MITM phishing toolkit space for multiple years.

To measure our classifier's performance when encountering a previously unknown MITM phishing toolkit, we iteratively remove each toolkit's data from our training set, leaving only the two remaining toolkits and known non-MITM phishing toolkit data. We then train with this new dataset, and test with the toolkit whose data we left out. This excluded data effectively acts as a new toolkit, previously unknown to the classifier. The results of this experiment are shown in Table 2C. We find that the accuracy of our classifier drops by less than 2% when testing on data from a completely unknown MITM phishing toolkit. As network architecture is constant across all MITM phishing toolkits, our classifier maintains consistently high performance regardless of the introduction of unfamiliar data.

3.6 PHOCA: MITM Phishing Toolkit Detection

Utilizing the previously described machine learning classifier, we develop a tool to automatically collect data on, and classify MITM phishing toolkits on the web. We call this tool PHOCA, after the Latin word for "seal." Seals are aquatic mammals known to hunt hidden prey using vibrations generated by their breathing. Similarly to this hunting technique, PHOCA can detect previously-hidden MITM phishing toolkits using features inherent to their nature, as opposed to visual-cues.

When provided either a URL or domain-name, PHOCA probes the desired web server to collect the previously mentioned network-level features. PHOCA then uses our trained classifier to determine if the web server is a MITM phishing toolkit. Using this tool, new training data can be easily generated for any future MITM phishing toolkit iteration. Additionally, PHOCA can be integrated into existing anti-phishing workflows to fingerprint active threats.

4 DISCOVERING MITM PHISHING SITES IN THE WILD

Using PHOCA, we conduct a large-scale search for MITM phishing toolkits in the wild. We seek to determine the online presence of these tools, and uncover patterns in their usage. This allows us to expose the source of phishing campaigns leveraging these tools, as well as targeted users and trademarks.

We start by designing and implementing a URL crawling infrastructure that visits thousands of potential phishing web pages each day, recording information about them, and classifying them as a MITM phishing toolkit or not.

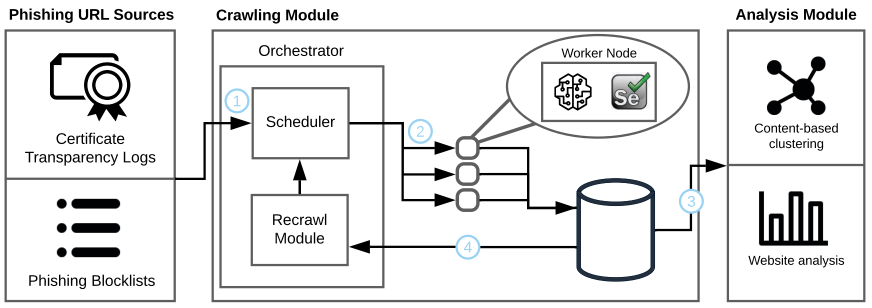


Figure 5: Architecture of framework used to collect network data on real world MITM phishing websites.

4.1 Phishing Website Crawling Infrastructure

We design a URL crawling infrastructure that visits phishing websites as they are created. Our crawlers collect information about each website and a label using PHOCA.

URL Collection

To obtain a comprehensive view into the MITM phishing ecosystem, we crawl URLs from popular open-source phishing databases (*Phishtank* [19] and *OpenPhish* [18]), as well as Certificate Transparency logs [4]. We utilize the Facebook Certificate Transparency Phishing API [9] to receive alerts for certificate registrations of impersonating domain names. Additionally, we supplement this source with our own Certificate Transparency log parser which uses regular expressions to search for combosquatting domains [40]. In total, we search for impersonating domains of 22 trademarks using Certificate Transparency (full list of trademarks can be found in Table **??** in the Appendix). We note that both Facebook's and our own Certificate Transparency log parsers do not classify MITM phishing websites, but rather search for domain names that appear to be impersonating known trademarks. We use all URLs and domains provided from each source as input to our crawling infrastructure to find MITM phishing websites deployed in the wild.

It is important to note that our reliance on Certificate Transparency and phishing lists excludes, by design, any test deployments of the evaluated MITM phishing toolkits by security analysts and researchers. Namely, we argue that setting up a subdomain or purchasing a domain name that matches the target site and deploying a phishing toolkit there, crosses the line between benign and malicious. Had we used Internet-wide scanning tools (such as ZMAP [35]) to identify MITM phishing toolkits, we would not be able to reliably differentiate between test deployments and deployments by attackers.

We also note that our use of Certificate Transparency logs limits us to only domain names rather than full URLs, which are typically available on phishing blocklists. However, since PHOCA uses network-level features to discover MITM phishing toolkits, we do not require access to phishing content. This is a strength of our detection technique, as we are intuitively fingerprinting the phishing web server rather than the phishing content, as is the case with traditional phishing detection.

Crawling Infrastructure

Figure 5 presents a high-level view of our URL crawling infrastructure. (1) Our queue-based system takes as input URLs from a number of sources, including phishing blocklists and impersonating domains found on the Certificate Transparency logs, and (2) dispatches one of our crawlers to collect data on the corresponding website in real time. Each crawler consists of two modules: a headless Selenium [21] browser, and a PHOCA worker. For each website encountered, we record the following information: *i)* HTML and screenshot of landing page, *ii)* TLS certificate offered to our browser, *iii)* original and redirected (if applicable) domain IP address, and *iv)* original and redirected (if applicable) classification.

(3) This data is forwarded to the analysis module which clusters web pages based on their content, assisting in the manual verification of the classifications made by our model. Furthermore, using information such as TLS certificates, web server IP addresses, and domain names allows us to uncover connections between seemingly independent phishing websites and cluster individual phishing sites into phishing campaigns.

(4) Our system also utilizes time-based re-queuing to record data on websites previously crawled in order to map the life cycle of MITM phishing websites over time. We make use of this when crawling domains from the Certificate Transparency logs. Since these domains are captured as soon as their certificates are created, there is a high probability that there will be no web server responding to requests at that moment. Thus, we re-crawl all such domains periodically following their initial recording. Moreover, we re-crawl all URLs classified as one of the phishing tools in order to measure how long these websites remain active after creation.

4.2 Experimental Evaluation and Results

MITM Phishing Toolkit Presence

We deployed our phishing website crawling infrastructure for 365 days from March 25, 2020 to March 25, 2021. This period was broken up into two phases. The first phase was an exploratory one which included URLs from both Certificate Transparency as well as the phishing URL databases OpenPhish and PhishTank. We used this phase to determine the most effective sources to capture MITM phishing websites.

During this first phase, we captured 17 MITM phishing websites from the URLs reported by OpenPhish and PhishTank, compared to the 189 captured from the domains reported by Facebook's Certificate Transparency API. We find that the highly targeted nature of attacks conducted using MITM phishing toolkits makes it difficult for user-curated blocklists to effectively report these websites in a timely manner. Furthermore, the cloaking abilities of MITM

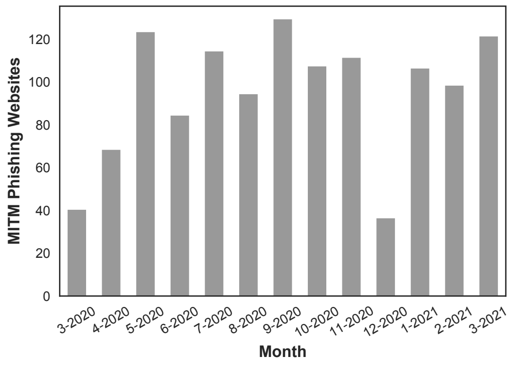


Figure 6: Number of MITM phishing websites identified each month of our data collection period.

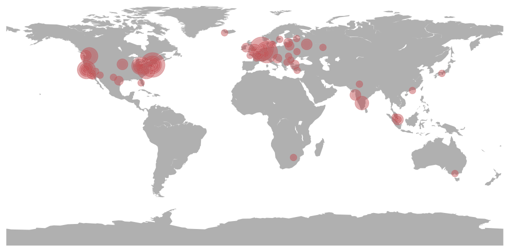


Figure 7: Geographic locations of MITM phishing websites. Larger circles indicate more IP addresses in a given area

phishing toolkits make it such that only victims possessing the tokenized URL will see the malicious content, while all other visitors are redirected to a benign website. This further complicates the task of identifying and blocklisting these websites as existing crawlers will never observe the malicious content.

Preemptively scanning potential phishing websites from sources such as Certificate Transparency is clearly the most effective way to detect attacks from these toolkits before damage is done. Thus, after our exploratory phase, we remove Phishtank and OpenPhish from our URL sources, opting to focus our infrastructure's resources on an additional Certificate Transparency source (to supplement Facebook's Certificate Transparency feed, as mentioned in Section 4.1). During this second phase, PHOCA labeled 6,875 sites as operated by MITM phishing toolkits, 849 from Facebook's Certificate Transparency API and 6,026 from our own Certificate Transparency Log parser. For brevity, we will refer to these two sources as *Certificate Transparency* for the remainder of this paper.

Upon completion of the second phase, we manually inspected the data collected on each website labeled as a MITM phishing toolkit to remove false positives. We did this by analyzing the screenshot, HTML, and classification data of each positively labeled phishing website and confirming that the content targets a popular trademark, and the network-level data matches the profile of a MITM phishing toolkit. In total, we identified 5,861 false positives from a total of 7,081 positive classifications, and promptly removed them from our dataset before conducting any further data analysis. Throughout our recording period, we classified 841,711 web pages, meaning PHOCA had a 0.6% false positive rate during our data collection period—approximately sixteen per day, on average.

Table 3: MITM phishing websites discovered per autonomous system

Autonomous System	IPs	Domains
Amazon.com, Inc.	162	136
DigitalOcean, LLC	160	386
Microsoft Corporation	62	165
Google LLC	37	61
Versatel Deutschland GmbH	15	1
Choopa, LLC	14	50
OVH SAS	13	38
Linode, LLC	9	40
HKT Limited	8	1
Other	150	354

Through close inspection of all false positives in our dataset, we find 5,298 belong to domain parking services, 5,158 of which belong to the domain parking service `sedo.com`. By analyzing the network timings and TLS fingerprints of `sedo.com` web pages, we infer that this service utilizes a reverse proxy infrastructure with similar network timing properties to MITM phishing toolkits. In practice, a web page classification system such as ours would benefit greatly from a pre-filtering step to remove common sources of misclassification. The addition of such a step that can filter websites that resolve to an IP address in the autonomous system of a domain parking service would leave only 563 false positives during our entire data collection period. This would result in an adjusted false positive rate of 0.067% or two false positives each day. This is in line with the results of our classifier in a laboratory setting (described in Section 3).

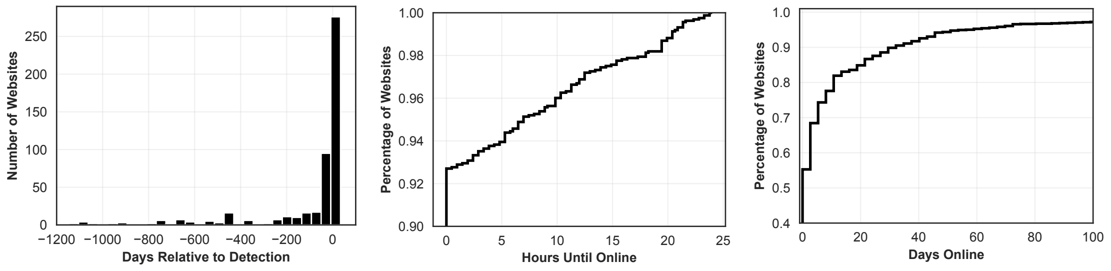
The remaining false positives consist of sites that we could not verify as being malicious based on our verification methodology. These consist of 230 empty or error web pages and 333 seemingly benign web pages that were included based on the similarity of their domain to a popular trademark that we followed. We theorize that these websites use a network architecture similar to that of MITM phishing toolkits with reverse proxy or caching servers. We note that these sites do not redirect visitors (e.g. for cloaking purposes), which could act as a second-level check to prevent false positives.

In total, we discovered 1,220 verified websites operated by MITM phishing toolkits over our entire data collection period, shown in Figure 6. We observe an upward trend in the number of MITM phishing toolkits discovered each month of our data collection period. This implies an increase in adoption of these toolkits by attackers—a trend we anticipate to continue into the future. We note a drop in the number of MITM phishing websites discovered in December 2020. We discovered that during this month, we received significantly fewer phishing URLs from our Certificate Transparency log API sources. However, we observe that the ratio of true positives to all phishing URLs in this month is consistent with all other months in this upward trend.

MITM Phishing Website Locations

We use the collected IP addresses to map each MITM phishing website we encountered, both in terms of its geographical location as well as the autonomous system in which the server hosting the toolkit belongs. This information allows us to identify hosting patterns as well as potential victims.

Figure 7 shows the geographic distribution of IP addresses associated with MITM phishing websites discovered by our crawling



(a) *(b)* *(c)*

Figure 8: (a) Days between MITM phishing domain registration and detection by our system (b) Hours before MITM phishing websites captured from Certificate Transparency logs go online (b) Days MITM phishing sites remain active

infrastructure. We find that while MITM phishing toolkits are in use around the world, most instances are located in North America and Europe. Further, these locations correspond to areas with large concentrations of hosting providers. We confirm this in Table 3 which shows the top autonomous systems in which MITM phishing websites were found, composed mostly of popular hosting providers.

The ability for attackers to quickly launch and remove cloud servers on hosting providers makes them a popular location for MITM phishing websites. In addition, since these toolkits attempt to impersonate real websites, being located on popular web hosting infrastructure could thwart security scanners searching for websites hosted in low-quality autonomous systems.

MITM Phishing Website Life Cycle

By utilizing WHOIS information, we are able to determine how long domains associated with MITM phishing websites were active, before they were discovered by our crawlers (Figure 8a). We find that most domains associated with MITM phishing websites were registered within the year prior to detection with 45.3% of domains being registered in the week of their detection.

Additionally, due to our infrastructure's re-crawling module, can determine the life cycle of MITM phishing websites from their creation to deletion. As mentioned in Section 4.1, when our data collection infrastructure receives a potential phishing domain from Certificate Transparency, it recrawls that domain every 3-6 hours (depending on the load of our system) for the following 2 days. Thus, we are able to determine how long it takes for a MITM phishing website to be ready to receive requests from victims after its certificate is created. Moreover, as our system continually recrawls all positively labeled MITM phishing websites, we are able to determine the amount of time these sites remain online after they are first observed. Figure 8b shows the number of hours it took MITM phishing websites in our dataset to come online after the creation of their TLS certificates. We find that all MITM phishing websites respond to requests within one day, with over 90% responding immediately.

Phishing campaigns utilizing traditional methods and tools are typically short lived, staying online for less than one day on average [52]. Given the fidelity of content that MITM phishing toolkits present to users in addition to their post-authentication operation and built-in evasion mechanisms (described in Section 3.2), we expect that the lifespan of such websites is higher than traditional phishing campaigns. Figure 8c shows the number of days MITM phishing websites in our dataset remain online after first discovery.

Table 4: Popular trademarks targeted by MITM phishing toolkits.

Brand	# Websites	Example Domain
Instagram	298	*m.logins-instagram.ga*
Google	249	*accounts.google-2fa.com*
Facebook	198	*sign-in.facebookes.com*
Outlook	92	*login.outlooks-mail.com*
Paypal	84	*paypalsecured.com*
Apple	76	*apple.icloud.com.sssl.host*
Twitter	63	*login.mobiletwitter.tk*
Coinbase	56	*googletag.coinbasel.com*
Yahoo	50	*yahoo.com.msg-inbox.ga*
Linkedin	41	*linkedin.com.securelogin.xyz*

We find that over 40% of MITM phishing websites in our dataset remain online for more than one day, with approximately 15% remaining online for over 20 days.

Targeted Brands and Phishing Campaigns

Through analysis of passive DNS data, we find that 339 (27%) domains associated with MITM phishing toolkits in our dataset are co-located on the same IP address as a benign domain, and 23 domains are co-located with at least one other domain tagged as malicious by the domain blocklists reported by VirusTotal. This indicates that attackers typically acquire dedicated infrastructure, or reuse existing malicious infrastructure, for their campaigns rather than compromise existing domains. This contrasts prior work on traditional phishing campaigns which report that half of all phishing domains resolve to IP addresses of legitimate websites [31].

Over the course of our study, we discovered 19 trademarks targeted by MITM phishing toolkits. However, we find that a subset of brands are disproportionately targeted—the top five most targeted attracting 67.1% of all MITM phishing websites in our dataset. Table 4 shows the number of MITM phishing websites discovered together with example impersonating domains targeting popular trademarks in our dataset. Furthermore, Figure 9 presents the distributions of impersonating domain types targeting the most popular trademarks we monitor. The domain names associated with MITM phishing toolkits we observed fall into three categories: combosquatting (e.g. `paypalhelp.com`), target embedding (e.g. login.paypal.com.attacker.com), and typosquatting (e.g. paypl.com). Domain distributions can vary greatly depending on the trademark, where domains impersonating Yahoo were almost entirely target

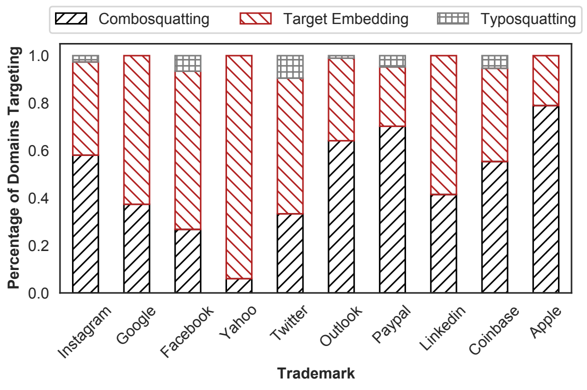


Figure 9: Types of phishing domain names targeting popular trademarks in our dataset.

embedding, while 70% of Paypal domains were combosquatting. Finally, Table 5 shows the distribution of top-level domains (TLDs) in our dataset. We observe that the domains with the `.com` TLD are the most common. Since gTLDs typically require substantial costs to register, their common presence in our dataset suggests that the discovered MTIM phishing toolkits were used in real (and most likely targeted) phishing attacks where believability of the domain is vital and therefore justify the increased domain-registration costs.

Phishing Blocklist Presence

By querying popular phishing blocklists, we discover that most MITM phishing websites our crawler discovered are missing from these lists. In total, only 43.7% of positively labeled URLs in our dataset are listed as malicious by at least one domain blocklist reported by VirusTotal [25]. Furthermore, it takes on average seven days *after* our crawlers discover a MITM phishing website, for these URLs to be labeled as such. This is substantially longer than traditional phishing campaigns, which previous work has shown are detected by blocklists after only nine hours [52]. Additionally, we find that only 18.9% of IP addresses associated with MITM phishing toolkits appear on at least one IP blocklist reported by VirusTotal. This suggests that attackers use fresh IP addresses and domains to launch attacks and quickly move before they are discovered.

MITM Phishing Toolkit Cloaking

Our results demonstrate that due to cloaking and their highly-targeted nature, the phishing websites supported by MITM phishing toolkits are able to remain hidden from the scanners that feed popular blocklists. This affords attackers with more time to inflict damage on a greater number of victims and decreased resource costs.

Our use of domain names from Certificate Transparency prevents us from obtaining the tokenized URLs, and in turn, seeing phishing content from Evilginx deployments (described in Section 3.2). Thus, we observed a diverse set of cloaking responses to requests towards MITM phishing websites in the wild. Of the 1,220 MITM websites discovered, 14.8% did not perform any cloaking, 19% redirected visitors to the legitimate website of the victim service (e.g. a phishing page targeting PayPal redirecting users to the real PayPal website), and 66.2% redirected visitors to arbitrary destinations, such as, pages served by `google.com`.

Table 5: Number of MITM phishing websites discovered per TLD

Rank	TLD	Domains	Rank	TLD	Domains
1	com	376	6	gq	59
2	ga	124	7	net	51
3	ml	97	8	xyz	50
4	tk	87	9	org	36
5	cf	84	10	Other	256

4.3 Case Study: MITM Phishing Attacks in an Enterprise Setting

To study the real-world effects of MITM phishing attacks and compare the performance of our detection infrastructure to a commercial phishing detection solution, we partnered with Palo Alto Networks (PAN). Their in-line vantage points to network communications and large enterprise-customer base provides detailed insights into real world phishing attacks. While PAN deploys various web scanners and state-of-the-art phishing detection techniques, we show that our solution adds a significant amount of exclusive detections to their phishing coverage.

We provided a list of all 1,220 MITM phishing hostnames to PAN to determine when each was active based on entries from their anti-phishing database, and to enrich their database with new phishing URLs. In total, we find that 57.6% of MITM phishing domains discovered by our infrastructure were labeled as either explicitly malicious or highly likely to be malicious by PAN scanners. Furthermore, of the domains listed as such, 15.1% were given their respective label at least one week after our infrastructure discovered it.

PAN researchers investigated the remaining URLs (42.4%) that were absent from their anti-phishing database and found one of the reasons to be cloaking mechanisms of the MITM phishing toolkits preventing their scanners from observing malicious content. In particular, they observed a campaign redirecting towards legitimate URLs of streaming services, either not being weaponized at the time of the analysis or not detected due to cloaking. Moreover, by using our tool, PAN found another two phishing hostnames targeting their users with a similar cloaking behavior. These results highlight that content-based phishing detection techniques can be thwarted by cloaking. However, our methodology mitigates these issues by focusing entirely on network-level features. After conclusion of this experiment, PAN is implementing PHOCA into their infrastructure in order to capture future MITM phishing toolkit instances.

In addition, we observe that enterprise users are currently being targeted by MITM phishing toolkits. PAN provided us with statistics on the number of their clients that visited each MITM phishing website. Over a 6 month period they captured 6,403 requests directed towards 260 of our identified MITM phishing websites, logged from 368 distinct firewall devices. On average, each MITM phishing website received 25 requests (as recorded by PAN's middleware), with the most popular site receiving 4,728 requests by their clients. We find that attacks from MITM phishing toolkits are prevalent in the wild and are currently affecting real users. Moreover, while MITM phishing toolkits by their nature are geared towards highly-targeted attacks, some real world attacks result in large numbers of users falling victim.

5 SERVER SIDE MITM PHISHING TOOLKIT FINGERPRINTING

In previous sections, we showed that it is possible to fingerprint deployments of MITM phishing toolkits from the point of view of a client. Even though this fingerprinting is valuable because it allows scanning tools to now identify these types of toolkits in the wild, the users who are targeted by these campaigns will not benefit from this fingerprinting until these sites are added to popular blocklists.

In this section, we explore the fingerprintability of these MITM phishing toolkits, from the perspective of the target web server. A web server that is able to differentiate between benign requests and those originating from a MITM phishing toolkit will be able to flag the latter and protect end users, even if these users are not aware that they are interacting with a MITM phishing server.

5.1 TLS Fingerprinting

Since data in the application layer is under complete control of the attacker, classic browser-fingerprinting methods will not suffice to determine the presence of a MITM phishing toolkit from the perspective of a targeted web server. For instance, as all JavaScript is executed on the victim's device, fingerprinting scripts would return information about the victim rather than the toolkit.

We therefore seek to fingerprint MITM phishing toolkits at the TLS layer of the network stack. These toolkits are built for the explicit purpose of phishing authentication details and do not make use of typical web-client software. As a result, the TLS stacks utilized to communicate with target servers are not the common stacks that web servers typically observe from their users. Additionally, since these toolkits forward all HTTP request headers to the target web server, there are discrepancies in the TLS fingerprint when compared to the reported device and browser in the User-Agent.

To determine the uniqueness of the TLS implementations of the MITM phishing toolkits studied, we implemented a web page that records the HTTP request headers, IP address, and *JA3 TLS Fingerprint* [10] of each client. JA3 fingerprints are created by concatenating each field of the TLS Client Hello packet and computing the hash of the resulting string. This produces a unique identifier which can be used to tag TLS implementations and identify clients.

To characterize the distribution of TLS stacks of regular users, we purchased 13,000 advertising impressions from a popular advertising service. For each impression, the user's browser connects to a server under our control over the HTTPS protocol, allowing us to obtain the aforementioned TLS fingerprint as well as the user's HTTP User-agent header. Our web page contained a simple message that was completely unrelated to our study. Specifically, the web page echoed CDC guidelines related to curbing the transmission of COVID-19, i.e., encouraging users to wash their hands and practice social distancing. We did not ask users to provide any input (PII or otherwise), did not offer downloads, and did not send them any cookies or JavaScript code during these interactions.

5.2 Server Side TLS Fingerprinting Results

Through this process, we recorded 163 JA3 TLS fingerprints of various clients, corresponding to 4,311 distinct HTTP User-Agents. The distribution of device platforms and browsers of our TLS fingerprinting dataset is presented in Table 6. Our dataset consists

Table 6: Distribution of TLS fingerprints collected by device platform and browser, as reported via User-Agent headers.

Platform	Versions	Browsers	Versions	Combinations
iOS	60	5	39	130
Mac OS	39	7	72	178
Android	37	5	135	460
ChromeOS	16	1	15	16
Windows	8	7	165	254
PlayStation	3	1	1	3
Linux	2	4	47	51
Totals	165	30	474	1092

of a wide range of device platforms, including mobile and desktop operating systems, and browsers. For each platform and browser type we also record a diverse distribution of versions. Overall, our dataset encompasses a large percentage of device platforms and browsers used today. We find that the JA3 fingerprints of the MITM phishing toolkits studied are unique in our dataset.

To reinforce our findings, we searched for the JA3 fingerprints of MITM phishing toolkits in the *ja3er.com* [11] fingerprint database. This database contains over 75 thousand unique fingerprints from a wide variety of platforms. We found 745 HTTP User-Agents that shared a TLS fingerprint with one of the three MITM phishing toolkits. However, we are only interested in finding collisions between the JA3 TLS fingerprints of MITM phishing toolkits and web browsers utilized by real users. We therefore filter this list of 745 HTTP User-Agents to remove any web bots.

Close inspection of all colliding User-Agents revealed a majority to be benign web bots utilizing the same Golang TLS libraries as MITM phishing toolkits. These web bots announce their identities in their User-Agent strings, often times with a link to a website explaining the purpose of the bot. The remaining User-Agents claimed to be popular browsers, but we determined these to be spoofed through manual verification of each User-Agent. We determined a User-Agent to be spoofed by either observing errors in the User-Agent string (e.g. misspelling of browser name), observing discrepancies in the reported browser and TLS fingerprint (i.e. the reported browser does not support a TLS feature claimed by the fingerprint), observing that the TLS fingerprint is an outlier compared to all other TLS fingerprints associated with the User-Agent, or manually recreating the JA3 fingerprint of the reported browser.

We conclude that the TLS fingerprints of MITM phishing toolkits are unique when compared to the fingerprints of popular browsers used by real web clients. It is therefore possible for a web server to distinguish the requests from MITM phishing toolkits from the benign requests of popular browsers with high accuracy, using only TLS fingerprints. Since the TLS fingerprints of these toolkits match only those of web bots utilizing the same Golang TLS libraries, it is safe for web servers to assume authentication requests from clients sharing one of these TLS fingerprints are suspicious, and should prompt closer inspection of all subsequent requests or further action from the user.

6 DISCUSSION

Due to the ubiquitous presence of online services in our lives, phishing campaigns remain a constant threat. Users who fall victim to these attacks face serious financial and personal repercussions due to the sensitive nature of stolen information. Furthermore, brands targeted by these attacks see a deterioration in their reputation among their user base, who may view a phishing campaign as a sign of insecure systems. MITM phishing toolkits magnify these issues by allowing attackers to launch highly sophisticated campaigns in which users are presented with web pages indistinguishable to those of the targeted brand. It is therefore of the utmost importance to develop tools and methodologies to defend against these attacks, in order to stop them before damage can be done.

6.1 Key Takeaways

- **MITM phishing toolkit fingerprinting:** The real-time traffic-proxying of MITM phishing toolkits that allows them to launch powerful phishing attacks, also exposes them to fingerprinting that is not available for traditional phishing techniques. In this paper, we demonstrated the effectiveness of using network-layer features to detect MITM phishing toolkits (Section 3), with the resulting classifier able to detect a large range of MITM phishing toolkits, including those utilizing cloaking techniques. Additionally, the responsibility of detecting these campaigns can be distributed to both sides of the tainted communication channel, thereby greatly increasing the probability of identifying malicious content early in its life cycle.
- **Classification in the wild:** Over the course of our longitudinal study, we discovered 1,220 MITM phishing toolkits targeting popular trademarks such as Google, Facebook, and Yahoo. Moreover, by collaborating with Palo Alto Networks, we identified that enterprise users are being targeted by MITM phishing toolkits.
- **Blindspot in phishing blocklists:** The cloaking mechanisms utilized by MITM phishing toolkits severely decrease the effectiveness of crowd-sourced blocklists (56.3% of the discovered URLs were missing from all evaluated blocklists). Phishing blocklist services must take a more proactive approach in discovering phishing content. We show that monitoring Certificate Transparency logs for impersonating domain names is a successful approach to uncovering otherwise hidden phishing websites.
- **Mitigations:** As we have discussed, the phishing content that victims receive from MITM phishing toolkits is directly from the targeted website. Thus, online services could include integrity checks within the web page source. This code could ensure the domain in the URL bar matches that of the real service, and reject authentication if it is not. However, as attackers have full control over application content, payload integrity cannot be ensured. If an attackers knows a particular service uses such application-layer integrity checks, they could simply remove this code prior to sending it to the victim. We do note however, that while it is possible for attackers to bypass these integrity checks, it is not trivial as online services can consistently change the signature of this code to thwart static analysis by attackers.

  As a more robust counter-measure, online services should simply use separate communication channels to complete 2FA. For instance, users could be sent a rendezvous URL through a second, secure communication channel, such as email. Users would then submit their 2FA code to the form located on this web page rather than the one presented by a MITM phishing toolkit. Similarly, Universal Two Factor (U2F) can be used to mitigate these attacks. As the generated key is bound to the domain of the intended online service, keys generated during authentication with a MITM phishing toolkit will be invalid.

6.2 Limitations

Our analysis should be considered alongside certain limitations. Since this is the first investigation of MITM phishing toolkits, when developing our classifier, we lacked real world ground-truth data from these types of toolkits. We remedied this by creating our own dataset, as described in Section 3.4. It is difficult, however, to create a completely representative dataset modeling discrepancies introduced by individual attackers, or the creation of new toolkits, from a laboratory setting. For instance, an attacker that places a MITM phishing toolkit behind an extra layer of redirection, such as a load balancer, will introduce additional packet RTT delays. Edge cases such as this require further training data to effectively identify. However, we show in Section 3.4 that training data can be quickly and easily generated to update our classifier to match such modifications. Moreover, as our classifier includes network timing features that are consistently present in reverse proxy-server deployments, it is agnostic to many modifications made by attackers.

Additionally, due to the overwhelming volume of TLS certificates registered and logged to Certificate Transparency, as well as resource limitations, we are unable to monitor all brands potentially targeted by MITM phishing toolkits. Rather, we monitored a subset of these brands based on their popularity and use of two-factor authentication. Implementations of our methodology by large institutions can expand monitoring to a larger subset of brands to discover more phishing websites.

Lastly, while we show that our fingerprinting technique is highly effective against MITM phishing toolkits, it is unable to discover traditional phishing websites. This is a strength of visual phishing detection compared to our approach. However, we note that PHOCA can be easily implemented into existing anti-phishing services, and should be used as an additional tool beside visual phishing detection, rather than a replacement to it.

6.3 Ethical Considerations and Responsible Disclosure

We took special care to ensure that PHOCA's probes are not intrusive and disruptive to the websites being analyzed. Each visit by our crawler results in a limited number of network requests ranging from initiating TCP and TLS handshakes to HTTP GET requests. Our crawler does not probe for vulnerabilities or use excessive server resources. Our only interaction with users was for determining the distribution of TLS stacks in the wild. As described in Section 5.1, our experiment collected the TLS Client-Hello message and the User-Agent header of each user, something that users volunteer to each and every website they visit on a daily basis. No PII or other user-provided information was requested and we did not make use of either stateful (e.g. cookies) or stateless (i.e. browser fingerprinting) tracking techniques.

In order to strengthen existing anti-phishing efforts, we have contacted a number of parties to disclose our findings. We have reached out to phishing blocklist services to share the URLs we discovered and ensure they make it onto widely-used blocklists. We also shared the information on the blind spots that currently exist in phishing blocklists as well as steps they can take to uncover MITM phishing websites. Finally, through our collaboration with Palo Alto Networks and their decision to adopt our technology, we are confident that employees of thousands of companies will be more protected against phishing attacks.

7 RELATED WORK

To the best of our knowledge, this work is the first to present a comprehensive study on MITM phishing toolkits, their fingerprintability, and potential defenses against them. In this section, we briefly discuss prior work on phishing ecosystem measurement and analysis, as well as network fingerprinting.

Phishing Ecosystem Analysis

Previous work has measured the overall lifespan of phishing websites to determine their effectiveness as well as the effectiveness of phishing blocklists. In 2020, Oest et al. proposed a framework to detect phishing against the infrastructure of a particular brand using HTTP referrer headers, allowing them to measure the complete lifespan of phishing web pages [52]. Han et al. studied the lifespan of phishing attacks deployed on honeypots, discovering that the average lifetime of phishing websites is eight days [36]. Sheng et al. studied the effectiveness of phishing blocklists, finding that the majority of phishing campaigns last less than two hours before detection [55]. Oest et al. examined the cloaking functionalities of phishing kits and how they affect the response time of phishing blocklist services [49, 51].

In this paper, we study a specific class of phishing toolkits that provide attackers with powerful evasion abilities due to the proxying of content live from the target website.

Phishing Attack Detection and Mitigation

Previous works have proposed detecting phishing websites based on the visual perception of web pages. This has been done through matching the perceptual features of phishing web pages to those of legitimate websites [27, 28, 46]. Visual differences in web pages of the same website have also been used to detect phishing content on compromised domains [34]. Prior work has also explored analyzing domain names and URLs for features indicative of phishing websites. To decrease the reaction time of phishing blocklists, efforts have been taken to detect the registration of phishing domain names in real time [37, 42, 43]. Similarly, prior work has proposed using features from URLs to classify phishing websites [32, 41, 53], as well as a combination of URL features and web page content [39, 44, 61].

Ulqinaku et al. demonstrated a system to mitigate 2FA phishing attacks by utilizing the user's smartphone to verify the URL in the user's browser [58]. When authenticating with an online service, an encrypted JavaScript payload must be decrypted using a key sent from the mobile device over Bluetooth and executed by the user's browser. The output of this script, a string representing the current URL, is sent back to the mobile device and verified before authentication can be completed.

In contrast to prior work, we propose techniques to detect MITM phishing websites using features independent of attacker controlled content from the perspectives of the client and targeted web server.

Proxy Server Fingerprinting

Attackers can use proxy servers to silently steal or modify data in transit. Thus, prior work has proposed methods to fingerprint proxy servers using the discrepancies they introduce. For instance, techniques have been presented to detect transparent forward and reverse proxies by analyzing transport and application layer responses to a set of probing requests [47, 60]. Further, analysis of network timing discrepancies has been used to determine the presence of HTTP reverse proxies in network communications [29, 59].

In this work, we propose a classifier that determines the presence of MITM phishing toolkits using a combination of network timing analysis with TLS fingerprinting. Timing analysis of packet RTTs is a robust method to fingerprint these toolkits, however including TLS fingerprinting strengthens our classifier, further increasing its robustness to attacker modifications.

8 CONCLUSION

MITM phishing toolkits magnify the damage caused due to phishing by enabling attackers to launch highly sophisticated campaigns, that appear visually indistinguishable to their victims. However, the aspect of these toolkits that make them as effective as they are, traffic proxying, is also their greatest weakness. Due to the network architecture of these malicious reverse proxy servers, discrepancies in network-level features can be used to infer their presence.

In this paper, we showed that it is possible to identify these tools at a network level, and proposed a classifier capable of detecting the presence of a MITM phishing toolkit with 99.9% accuracy. We also create a fingerprinting tool, called PHOCA, to automatically collect data on, and detect MITM phishing toolkits on the web. Using PHOCA, we monitored popular phishing blocklists and Certificate Transparency logs for 365 days, discovering 1,220 websites powered by MITM phishing toolkits targeting major brands. We found that these websites are hosted on dedicated malicious servers and are largely absent from popular URL blocklists. Through our collaboration with Palo Alto Networks, we demonstrate the real-world presence of MITM phishing toolkits, observing 6,403 customer requests towards 260 of these toolkit deployments over a six-month period.

Finally, next to client-side fingerprinting, we also presented a methodology that targeted brands can use to detect malicious requests originating from MITM phishing toolkits using TLS fingerprinting. From a survey of 4,311 distinct device User-Agents, the TLS fingerprints of MITM phishing toolkits are unique, and thus fingerprintable from network requests alone.

Availability: To help security researchers and maintainers of phishing blocklists, we are open-sourcing PHOCA and our curated datasets. Moreover, we are making available our code for creating testing environments across globally-distributed public clouds (described in Section 3.4) so future research can obtain timing measurements from all current and future MITM phishing tools. All code and data can be found at https://catching-transparent-phish.github.io

Acknowledgements: We thank our shepherd Sooel Son and the anonymous reviewers for their helpful feedback. This work was supported by the Office of Naval Research (ONR) under grant N00014-20-1-2720, as well as by the National Science Foundation (NSF) under grants CMMI-1842020, CNS-1813974, CNS-1941617, and CNS-2126654.

REFERENCES

[1] 2021. Amazon Web Services. https://aws.amazon.com.
[2] 2021. Apache Traffic Server. https://trafficserver.apache.org.
[3] 2021. Apache Web Server. https://apache.org.
[4] 2021. Certificate Transparency. https://certificate-transparency.org.
[5] 2021. CloudFlare. https://cloudflare.com.
[6] 2021. CredSniper. https://github.com/ustayready/CredSniper.
[7] 2021. Digital Ocean. https://digitalocean.com.
[8] 2021. Evilginx. https://github.com/kgretzky/evilginx2.
[9] 2021. Facebook Certificate Transparency API. https://developers.facebook.com/docs/certificate-transparency.
[10] 2021. JA3. https://github.com/salesforce/ja3.
[11] 2021. JA3er. https://ja3er.com.
[12] 2021. Let's Encrypt. https://letsencrypt.org.
[13] 2021. Linode. https://linode.com.
[14] 2021. Modlishka. https://github.com/drk1wi/Modlishka.
[15] 2021. Muraena. https://github.com/muraenateam/muraena.
[16] 2021. Necrobrowser. https://github.com/muraenateam/necrobrowser.
[17] 2021. Nginx. https://nginx.com.
[18] 2021. Openphish. https://openphish.com.
[19] 2021. Phishtank. https://phishtank.com.
[20] 2021. Reelphish. https://github.com/fireeye/ReelPhish.
[21] 2021. Selenium. https://selenium.dev.
[22] 2021. Squid Proxy Server. http://squid-cache.org.
[23] 2021. TLS Prober. https://github.com/WestpointLtd/tls_prober.
[24] 2021. TOR. https://torproject.org.
[25] 2021. VirusTotal. https://virustotal.com.
[26] 2021. VirusTotal Evilginx Analysis Results. https://www.virustotal.com/gui/file/35636566f7ce11d44c2acf6ce6dca00b730b39710e82000a0e92b4af4a75e52c/detection.
[27] Sahar Abdelnabi, Katharina Krombholz, and Mario Fritz. 2020. VisualPhishNet: Zero-Day Phishing Website Detection by Visual Similarity. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*.
[28] Moruf A Adebowale, Khin T Lwin, Erika Sanchez, and M Alamgir Hossain. 2019. Intelligent web-phishing detection and protection scheme using integrated features of Images, frames and text. *Expert Systems with Applications* 115 (2019), 300–313.
[29] Daniel R Alexander. 2015. *Inferring the Presence of Reverse Proxies Through Timing Analysis*. Technical Report. Naval Postgraduate School Monterey CA.
[30] Manos Antonakakis, Roberto Perdisci, Wenke Lee, Nikolaos Vasiloglou, and David Dagon. 2011. Detecting Malware Domains at the Upper DNS Hierarchy.. In *Proceedings of the 10th USENIX Security Symposium (USENIX Security)*, Vol. 11. 1–16.
[31] APWG. 2017. *Global Phishing Survey: Trends and Domain Name Use in 2016*. Technical Report.
[32] Aaron Blum, Brad Wardman, Thamar Solorio, and Gary Warner. 2010. Lexical feature based phishing URL detection using online learning. In *Proceedings of the 3rd ACM Workshop on Artificial Intelligence and Security*. 54–60.
[33] Davide Canali, Marco Cova, Giovanni Vigna, and Christopher Kruegel. 2011. Prophiler: a fast filter for the large-scale detection of malicious web pages. In *Proceedings of the 20th International Conference on World Wide Web (WWW)*. ACM, 197–206.
[34] Igino Corona, Battista Biggio, Matteo Contini, Luca Piras, Roberto Corda, Mauro Mereu, Guido Mureddu, Davide Ariu, and Fabio Roli. 2017. Deltaphish: Detecting phishing webpages in compromised websites. In *Proceedings of the 2017 European Symposium on Research in Computer Security*. Springer, 370–388.
[35] Zakir Durumeric, Eric Wustrow, and J Alex Halderman. 2013. ZMap: Fast Internet-wide scanning and its security applications. In *22nd USENIX Security Symposium*. 605–620.
[36] Xiao Han, Nizar Kheir, and Davide Balzarotti. 2016. Phisheye: Live monitoring of sandboxed phishing kits. In *Proceedings of the 2016 SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 1402–1413.
[37] Shuang Hao, Alex Kantchelian, Brad Miller, Vern Paxson, and Nick Feamster. 2016. PREDATOR: proactive recognition and elimination of domain abuse at time-of-registration. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 1568–1579.
[38] Luca Invernizzi, Kurt Thomas, Alexandros Kapravelos, Oxana Comanescu, Jean-Michel Picod, and Elie Bursztein. 2016. Cloak of visibility: Detecting when machines browse a different web. In *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 743–758.
[39] Ankit Kumar Jain and Brij B Gupta. 2018. Towards detection of phishing websites on client-side using machine learning based approach. *Telecommunication Systems* 68, 4 (2018), 687–700.
[40] Panagiotis Kintis, Najmeh Miramirkhani, Charles Lever, Yizheng Chen, Rosa Romero-Gómez, Nikolaos Pitropakis, Nick Nikiforakis, and Manos Antonakakis. 2017. Hiding in plain sight: A longitudinal study of combosquatting abuse. In *Proceedings of the 2017 SIGSAC Conference on Computer and Communications Security*. ACM, 569–586.
[41] Anh Le, Athina Markopoulou, and Michalis Faloutsos. 2011. Phishdef: Url names say it all. In *Proceedings of the 2011 IEEE INFOCOM*. IEEE, 191–195.
[42] Xueni Li, Guanggang Geng, Zhiwei Yan, Yong Chen, and Xiaodong Lee. 2016. Phishing detection based on newly registered domains. In *2016 IEEE international conference on big data (big data)*. IEEE, 3685–3692.
[43] Samuel Marchal, Jérôme François, Thomas Engel, et al. 2012. Proactive discovery of phishing related domain names. In *International Workshop on Recent Advances in Intrusion Detection*. Springer, 190–209.
[44] Anutthamaa Martin, Na Anutthamaa, M Sathyavathy, Marie Manjari Saint Francois, Dr V Prasanna Venkatesan, et al. 2011. A framework for predicting phishing websites using neural networks. *arXiv preprint arXiv:1109.1074* (2011).
[45] S Matthew and Geoffrey Xie. 2013. *Fingerprinting Reverse Proxies Using Timing Analysis of TCP Flows*. Technical Report. Naval Postgraduate School Monterey CA.
[46] Eric Medvet, Engin Kirda, and Christopher Kruegel. 2008. Visual-similarity-based phishing detection. In *Proceedings of the 4th international conference on Security and privacy in communication netowrks*. 1–6.
[47] Antonio Nappa, Rana Faisal Munir, Irfan Khan Tanoli, Christian Kreibich, and Juan Caballero. 2016. RevProbe: detecting silent reverse proxies in malicious server infrastructures. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*. 101–112.
[48] Terry Nelms, Roberto Perdisci, Manos Antonakakis, and Mustaque Ahamad. 2016. Towards measuring and mitigating social engineering software download attacks. In *Proceedings of the 25th USENIX Security Symposium (USENIX Security)*. 773–789.
[49] Adam Oest, Yeganeh Safaei, Adam Doupé, Gail-Joon Ahn, Brad Wardman, and Kevin Tyers. 2019. Phishfarm: A scalable framework for measuring the effectiveness of evasion techniques against browser phishing blacklists. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1344–1361.
[50] Adam Oest, Yeganeh Safei, Adam Doupé, Gail-Joon Ahn, Brad Wardman, and Gary Warner. 2018. Inside a phisher's mind: Understanding the anti-phishing ecosystem through phishing kit analysis. In *2018 APWG Symposium on Electronic Crime Research (eCrime)*. IEEE, 1–12.
[51] Adam Oest, Yeganeh Safei, Adam Doupé, Gail-Joon Ahn, Brad Wardman, and Gary Warner. 2018. Inside a Phisher's Mind: Understanding the Anti-Phishing Ecosystem Through Phishing Kit Analysis. In *Proceedings of the 2018 APWG Symposium on Electronic Crime Research (eCrime)*. IEEE, 1–12.
[52] Adam Oest, Penghui Zhang, Brad Wardman, Eric Nunes, Jakub Burgis, Ali Zand, Kurt Thomas, Adam Doupé, and Gail-Joon Ahn. 2020. Sunrise to Sunset: Analyzing the End-to-end Life Cycle and Effectiveness of Phishing Attacks at Scale. In *Proceedings of the 29th USENIX Security Symposium (USENIX Security)*.
[53] Pawan Prakash, Manish Kumar, Ramana Rao Kompella, and Minaxi Gupta. 2010. Phishnet: predictive blacklisting to detect phishing attacks. In *2010 Proceedings IEEE INFOCOM*. IEEE, 1–5.
[54] Babak Rahbarinia, Roberto Perdisci, and Manos Antonakakis. 2015. Segugio: Efficient behavior-based tracking of malware-control domains in large ISP networks. In *Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. IEEE, 403–414.
[55] Steve Sheng, Brad Wardman, Gary Warner, Lorrie Cranor, Jason Hong, and Chengshan Zhang. 2009. An Empirical Analysis of Phishing Blacklists. (2009).
[56] Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna. 2010. Detecting spammers on social networks. In *Proceedings of the 26th Annual Computer Security Applications Conference*. 1–9.
[57] Kurt Thomas, Frank Li, Ali Zand, Jacob Barrett, Juri Ranieri, Luca Invernizzi, Yarik Markov, Oxana Comanescu, Vijay Eranti, Angelika Moscicki, et al. 2017. Data Breaches, Phishing, or Malware? Understanding the Risks of Stolen Credentials. In *Proceedings of the 2017 SIGSAC Conference on Computer and Communications Security*. ACM, 1421–1434.
[58] Enis Ulqinaku, Daniele Lain, and Srdjan Capkun. 2019. 2FA-PP: 2nd Factor Phishing Prevention. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*. 60–70.
[59] Matthew S Weant. 2013. *Fingerprinting reverse proxies using timing analysis of TCP flows*. Technical Report. Naval Postgraduate School Monterey CA.
[60] Nicholas Weaver, Christian Kreibich, Martin Dam, and Vern Paxson. 2014. Here be web proxies. In *Proceedings of the International Conference on Passive and Active Network Measurement*. Springer, 183–192.
[61] Colin Whittaker, Brian Ryner, and Marria Nazif. 2010. Large-scale automatic classification of phishing pages. In *Proceedings of the 17th Network and Distributed System Security Symposium (NDSS)*.

9 APPENDIX

```
proxy_hosts:
  - {phish_sub: '', orig_sub: '', domain: 'twitter.com',
        session: true, is_landing: true}
  - {phish_sub: 'abs', orig_sub: 'abs',
        domain: 'twimg.com'}
  - {phish_sub: 'api', orig_sub: 'api',
        domain: 'twitter.com'}
sub_filters: []
auth_tokens:
  - domain: '.twitter.com'
    keys: ['kdt','_twitter_sess','twid','auth_token']
credentials:
  username:
    key: 'session\[username_or_email\]'
    search: '(.*)'
    type: 'post'
  password:
    key: 'session\[password\]'
    search: '(.*)'
    type: 'post'
login:
  domain: 'twitter.com'
  path: '/login'
js_inject:
```

Listing 1: Example configuration file for Evilginx specifying options for a Twitter phishing web page

Table 7: Full feature set of our 2FA phishing toolkit classifier

Feature Name	Feature Type
Network Timing Features	
TCP SYN/ACK RTT	Numeric
TLS Client Hello RTT	Numeric
Malformed TLS Client Hello RTT	Numeric
TLS Handshake Timing	Numeric
HTTP GET Request Timing	Numeric
HTTP GET Request w/o Host Header Timing	Numeric
Malformed HTTP GET Request Timing	Numeric
HTTPS GET Request Timing	Numeric
HTTPS GET Request w/o Host Header Timing	Numeric
Malformed HTTPS GET Request Timing	Numeric
TCP SYN/ACK to HTTP GET Request Timing Ratio	Numeric
TCP SYN/ACK to Malformed HTTP GET Request Timing Ratio	Numeric
TCP SYN/ACK to Malformed HTTPS Get Request Timing Ratio	Numeric
Malformed to Valid HTTPS GET Request Timing Ratio	Numeric
TLS Versions Supported	
SSLv2	Binary
SSLv3	Binary
TLSv1	Binary
TLSv1.2	Binary
TLSv1.3	Binary
TLS Library	
Apache Tomcat 7.0.53	Numeric
Apache Tomcat 7.0.54	Numeric
... 177 more TLS libraries ...	
openssl 1.0.2a default source build	Numeric
wolfSSL 3.4.0	Numeric

Table 8: Full set of trademarks our system received impersonating domains for from Certificate Transparency logs

Domain	
Email	Google, Yahoo, Outlook, Protonmail
Financial	Paypal, Chase, Wells Fargo, Bank of America
	HSBC, Citi Bank, Captial One, American Express
Social Media	Facebook, Instagram, Twitter, Linkedin
Web Services	Dropbox, Amazon, Github, Office 365
Technology	Apple, Microsoft